3. domaća zadaća iz Formalne verifikacije programske potpore

Implementacija SAT-rješavača

1. (12 bodova) Implementirajte osnovni DPLL SAT-algoritam u programskom jeziku po izboru.

Napomena: zbog brzine izvođenja preporuča se implementacija u jezicima C, C++, C# ili Java.

Detaljniji opis zadatka:
Vaš program treba prihvatiti unos u CNF-verziji DIMACS formata. Kod takvog unosa, ulazna datoteka je formula u konjunkcijskoj normalnoj formi (CNF), gdje je svaka klauzula dana u zasebnom retku kao popis indeksa koji označavaju propozicijske varijable, dok negativni indeks označava negiranu propozicijsku varijablu. Za detalje oko DIMACS formata pogledajte datoteku `satformat.pdf`
Ako je ulazna CNF-formula zadovoljiva, vaš SAT-rješavač treba kao rezultat proizvesti dodjelu svim varijablama (u svakom retku jedna dodjela, npr. `v1 = true`) tako da se cijela formula evaluira u istinito (`true`) pod tom dodjelom.
Ako je ulazna formula nezadovoljiva, vaš SAT-rješavač treba jednostavno reći `false`.
Odgovor SAT-rješavača treba pohraniti u izlaznu datoteku istoga naziva kao i problem koji se učitao, ali s ekstenzijom „`.txt`“.
Implementaciju osnovnog DPLL SAT-algoritma potrebno je ponuditi prema opisu algoritma danom na 9. predavanju.

2. (8 bodova) Kao nadogradnju osnovne implementacije, potrebno je ponuditi implementaciju najmanje dva proširenja osnovnog DPLL SAT-algoritma.

Napomena: moguće je ostvariti i bonus bodove za implementaciju više proširenja i to po četiri boda po uspješno implementiranom dodatnom proširenju (bez obzira na broj proširenja najviše je moguće ostvariti **8 bonus bodova**)

Detaljniji opis zadatka:
Postoje razna proširenja osnovnog DPLL-algoritma koja omogućuju SAT-rješavačima da postignu izvrsne rezultate u praksi. Ovdje je dan neiscrpan popis mogućih proširenja osnovnog DPLL algoritma (niste ograničeni ovim popisom):

- heuristika za odabir redoslijeda literala za grananje (npr. VSIDS),
- učenje klauzula nakon konflikta (engl. *Conflict-Driven Clause Learning*, CDCL),
- graf implikacija (engl. *implication graph*) s nekronološkim povratkom (engl. *backjumping*),
- heuristika za odbacivanje naučenih klauzula (npr. *scheduled lazy deletion*)
- heuristika za odabir ponovnog pokretanja pretraživanja drugim redoslijedom (uz zadržavanje naučenih klauzula) (engl. *restart*)
- specijalizirane strukture podataka za brze operacije (npr. za propagaciju Booleovih ograničenja)
- paralelizacija SAT-rješavača na više jezgri (engl. *parallel SAT-solvers*)
- korištenje rezolucijskog pravila negdje u postupku rasuđivanja (npr. kao kod MiniSAT-a)

- tehnike za pojednostavljivanje formule prije (*preprocessing*) i tijekom obrade (*inprocessing*), npr. *bounded variable elimination, hidden literal elimination, SAT sweep*.

Na predavanjima ste dobili smjernice oko mnogih ovih proširenja. Dodatna preporučena literatura je knjiga:

Armin Biere, Marijn Heule, Hans van Maaren and Toby Walsh (Eds.), "Handbook of Satisfiability", 2nd Ed., IOS Press, 2021.

koja je dostupna u Moodleu predmeta (nije za daljnju distribuciju).

SAT-problemi

Za isprobavanje uspješnosti implementacije SAT-rješavača studenti se trebaju poslužiti sa 7 SAT-problema priloženih ovoj zadaći. To su:

1. `uf20-01.cnf` – random SAT-problem – 20 varijabli, 91 klauzula, **zadovoljivo**.
Detaljniji opis: https://www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/RND3SAT/descr.html

2. `uuf50-01.cnf` – random SAT problem – 50 varijabli, 218 klauzula, **nije zadovoljivo**.
Detaljniji opis: https://www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/RND3SAT/descr.html

3. `flat50-1.cnf` – SAT-kodiran problem bojanja grafova – 150 varijabli, 545 klauzula, **zadovoljivo**.
Detaljniji opis: https://www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/GCP/descr.html

4. `medium.cnf` – SAT-kodiran problem planiranja "Blocks World" – 116 varijabli, 953 klauzula, **zadovoljivo**.
Detaljniji opis:
https://www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/PLANNING/BlocksWorld/descr.html

5. `hanoi4.cnf` – SAT-kodiran problem hanojskih tornjea – 718 varijabli, 4934 klauzule, **zadovoljivo**.
Detaljniji opis:
https://www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/DIMACS/HANOI/descr.html

6. `bf0432-007.cnf` – analiza kvarova strujnih krugova – 1040 varijabli, 3668 klauzula, **nije zadovoljivo**.
Detaljniji opis:
https://www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/DIMACS/BF/descr.html

7. `hole7.cnf` – SAT-kodiran problem "Pigeon hole" – 56 varijabli, 204 klauzule, **nije zadovoljivo**.
Detaljniji opis:
https://www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/DIMACS/PHOLE/descr.html

Osim navedenih skupova podataka, dodatni skupovi za isprobavanje uspješnosti implementacije SAT-rješavača mogu se preuzeti na nekoj od poveznica:

Benchmarks: https://www.cs.ubc.ca/~hoos/SATLIB/benchm.html
SAT-natjecanja: http://www.satcompetition.org/

Kolokviranje domaće zadaće

Kao pripremu za kolokviranje domaće zadaće potrebno je da studenti izvrte na svojoj implementaciji SAT-rješavača svih 7 dostupnih problema te da u nekoj tablici zapišu vremena trajanja izvršavanja pojedinih problema, po mogućnosti na konfiguraciji (laptopu) na kojoj će i demonstrirati rad programa. U slučaju da neki problem nije bio riješiv u razumnom vremenu (nekoliko sati), to je potrebno i navesti u tablici.

Na kolokviranje je potrebno donijeti laptop s vlastitom implementacijom SAT-rješavača.

Za svako proširenje koje implementirate trebate moći opisati tijekom kolokviranja ove zadaće što ona točno čini i kako je poboljšala izvorni DPLL-algoritam. Također, kako biste postigli što više bodova, na kolokviranju trebate demonstrirati različito (po mogućnosti bolje) vrijeme izvođenja uz korištenje svakog pojedinog poboljšanja. Pritom program treba omogućiti parametrizaciju toga koja proširenja koristi, a koja ne.

Nastavnici će provjeriti uspješnost implementacije (uz razgovor o implementaciji sa studentom) i dodijeliti odgovarajući broj bodova. Nakon kolokviranja, nastavnici će preuzeti implementaciju od studenta na prijenosni USB medij.